Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

Лабораторная работа №3 по курсу

«Операционные системы»

Группа: М8О-209БВ-24

Студент: Андреев М.В

Преподаватель: Миронов Е.С.

Оценка: _____

Дата: 26.11.25

Москва, 2025

Постановка задачи

Вариант 19.

Составить и отладить программу на языке Си, осуществляющую работу с процессами и взаимодействие между ними в одной из двух операционных систем. В результате работы программа (основной процесс) должен создать для решение задачи один или несколько дочерних процессов. Взаимодействие между процессами осуществляется через системные сигналы/события и/или через отображаемые файлы (memory-mapped files).

Необходимо обрабатывать системные ошибки, которые могут возникнуть в результате работы.

Общий метод и алгоритм решения

Использованные системные вызовы:

Системные вызовы для работы с разделяемой памятью:

• shm_open(name, oflag, mode) - создает или открывает объект POSIX разделяемой памяти с указанным именем, флагами и правами доступа

• ftruncate(fd, length) - устанавливает размер объекта разделяемой памяти до указанной длины

• mmap(addr, length, prot, flags, fd, offset) - отображает объект разделяемой памяти в адресное пространство процесса для прямого доступа через указатели

• munmap(addr, length) - удаляет отображение области памяти из адресного пространства процесса

• shm_unlink(name) - удаляет объект разделяемой памяти из системы по имени

Системные вызовы для работы с процессами:

• fork() - создает новый дочерний процесс как точную копию родительского процесса

• execl(path, arg0, arg1, ..., NULL) - заменяет образ текущего процесса новым исполняемым файлом с передачей аргументов командной строки

• waitpid(pid, status, options) - ожидает завершения указанного дочернего процесса и получает его статус завершения

• exit(status) - завершает выполнение текущего процесса с указанным кодом возврата

Алгоритм работы программы:

1. Инициализация родительского процесса

- Запрос имен разделяемой памяти и выходных файлов у пользователя
- Создание двух объектов разделяемой памяти через shm_open()
- Установка размеров памяти через ftruncate()
- Отображение памяти в адресное пространство через mmap()

- Инициализация структур данных нулями через memset()

2. Создание дочерних процессов

- Создание первого дочернего процесса через fork()
- Запуск программы child с параметрами через execl()
- Создание второго дочернего процесса аналогично
- Дочерние процессы подключаются к существующей разделяемой памяти

3. Взаимодействие через разделяемую память

- Родитель принимает строки от пользователя
- Вероятностная фильтрация: 80% строк → Child1, 20% строк → Child2
- Копирование строк в разделяемую память с установкой флага data_ready = 1
- Дочерние процессы отслеживают флаг data_ready, обрабатывают строки (удаляют гласные)
- Запись результатов в соответствующие выходные файлы
- Сброс флага data_ready = 0 после обработки

4. Завершение работы

- При команде "QUIT" установка флага process_complete = 1
- Ожидание завершения дочерних процессов через waitpid()
- Освобождение ресурсов: munmap(), close(), shm_unlink()

Код программы

parent.c

```c
#include "os_utils.h"

int create_shared_memory(const char* name) {
    int fd = shm_open(name, O_CREAT | O_RDWR, 0666);
    if (fd == -1) {
        perror("shm_open");
        return -1;
    }

    if (ftruncate(fd, sizeof(shared_data_t)) == -1) {
        perror("ftruncate");
        close(fd);
        return -1;
    }

    return fd;
}

int main() {
    char shm_name1[MAX_FILENAME_LENGTH];
    char shm_name2[MAX_FILENAME_LENGTH];
    char output_name1[MAX_FILENAME_LENGTH];
```

```
    char output_name2[MAX_FILENAME_LENGTH];

    shared_data_t *shared1, *shared2;
    int shm_fd1, shm_fd2;
    pid_t pid1, pid2;

    StatusCode status = STATUS_OK;

    srand(time(NULL));

    // Ввод имен для разделяемой памяти и выходных файлов
    printf("Введите имя shared memory для child1: ");
    if (fgets(shm_name1, sizeof(shm_name1), stdin) == NULL) {
        return INVALID_INPUT;
    }
    shm_name1[strcspn(shm_name1, "\n")] = '\0';

    printf("Введите имя файла вывода для child1: ");
    if (fgets(output_name1, sizeof(output_name1), stdin) == NULL) {
        return INVALID_INPUT;
    }
    output_name1[strcspn(output_name1, "\n")] = '\0';

    printf("Введите имя shared memory для child2: ");
    if (fgets(shm_name2, sizeof(shm_name2), stdin) == NULL) {
        return INVALID_INPUT;
    }
    shm_name2[strcspn(shm_name2, "\n")] = '\0';

    printf("Введите имя файла вывода для child2: ");
    if (fgets(output_name2, sizeof(output_name2), stdin) == NULL) {
        return INVALID_INPUT;
    }
    output_name2[strcspn(output_name2, "\n")] = '\0';

    // Создание разделяемой памяти
    shm_fd1 = create_shared_memory(shm_name1);
    shm_fd2 = create_shared_memory(shm_name2);

    if (shm_fd1 == -1 || shm_fd2 == -1) {
        return MMAP_ERROR;
    }

    // Отображение разделяемой памяти
    shared1 = mmap(NULL, sizeof(shared_data_t), PROT_READ | PROT_WRITE, MAP_SHARED,
shm_fd1, 0);
    shared2 = mmap(NULL, sizeof(shared_data_t), PROT_READ | PROT_WRITE, MAP_SHARED,
shm_fd2, 0);

    if (shared1 == MAP_FAILED || shared2 == MAP_FAILED) {
        perror("mmap");
        return MMAP_ERROR;
    }
```

```
    // Инициализация разделяемой памяти
    memset(shared1, 0, sizeof(shared_data_t));
    memset(shared2, 0, sizeof(shared_data_t));

    // Создание первого дочернего процесса
    pid1 = fork();
    if (pid1 == -1) {
        perror("fork");
        status = FORK_ERROR;
        goto cleanup;
    }

    if (pid1 == 0) {
        // Дочерний процесс 1
        execl("./child", "child", shm_name1, output_name1, NULL);
        perror("execl");
        exit(3);  // EXEC_ERROR = 3
    }

    // Создание второго дочернего процесса
    pid2 = fork();
    if (pid2 == -1) {
        perror("fork");
        status = FORK_ERROR;
        goto cleanup;
    }

    if (pid2 == 0) {
        // Дочерний процесс 2
        execl("./child", "child", shm_name2, output_name2, NULL);
        perror("execl");
        exit(3);  // EXEC_ERROR = 3
    }

    printf("Родительский процесс начался\n");
    printf("Введите строки (для выхода введите QUIT):\n");

    char buffer[MAX_LINE_LENGTH];
    while (fgets(buffer, sizeof(buffer), stdin) != NULL) {
        int len = strlen(buffer);
        if (len > 0 && buffer[len - 1] == '\n') {
            buffer[len - 1] = '\0';
            len--;
        }

        if (strcmp(buffer, "QUIT") == 0) {
            break;
        }

        // Вероятностная отправка (80% - child1, 20% - child2)
        int random_percent = rand() % 100;
        shared_data_t* target_shared;
        const char* target_name;
```

```
        if (random_percent < 80) {
            target_shared = shared1;
            target_name = "Child1";
        } else {
            target_shared = shared2;
            target_name = "Child2";
        }

        printf("Отправлено в %s (вероятность: %d%%)\n", target_name, random_percent);

        // Копирование данных в разделяемую память
        strncpy(target_shared->data, buffer, MAX_LINE_LENGTH - 1);
        target_shared->data_ready = 1;

        // Ожидание обработки дочерним процессом
        while (target_shared->data_ready == 1) {
            usleep(1000); // Небольшая пауза
        }
    }

    // Сигнал дочерним процессам о завершении
    shared1->process_complete = 1;
    shared2->process_complete = 1;

    // Ожидание завершения дочерних процессов
    waitpid(pid1, NULL, 0);
    waitpid(pid2, NULL, 0);

    printf("Родительский и дочерние процессы успешно завершены\n");

cleanup:
    // Освобождение ресурсов
    if (shared1 != MAP_FAILED) munmap(shared1, sizeof(shared_data_t));
    if (shared2 != MAP_FAILED) munmap(shared2, sizeof(shared_data_t));
    if (shm_fd1 != -1) {
        close(shm_fd1);
        shm_unlink(shm_name1); // удаление объекта
    }
    if (shm_fd2 != -1) {
        close(shm_fd2);
        shm_unlink(shm_name2);
    }

    return status;
}
```

child.c

```
#include "os_utils.h"

void remove_vowels(char* str) {
    if (!str) {
        return;
    }
```

```
    int write_idx = 0;
    for (int read_idx = 0; str[read_idx] != '\0'; read_idx++) {
        char c = str[read_idx];
        char lower_c = tolower((unsigned char)c);
        if (lower_c != 'a' && lower_c != 'e' && lower_c != 'i' &&
            lower_c != 'o' && lower_c != 'u' && lower_c != 'y') {
            str[write_idx++] = c;
        }
    }
    str[write_idx] = '\0';
}

int main(int argc, char* argv[]) {
    if (argc != 3) {
        fprintf(stderr, "Использование: %s <shm_name> <output_file>\n", argv[0]);
        return INVALID_INPUT;
    }

    const char* shm_name = argv[1];
    const char* output_name = argv[2];

    // Открытие разделяемой памяти
    int shm_fd = shm_open(shm_name, O_RDWR, 0666);
    if (shm_fd == -1) {
        perror("shm_open");
        return MMAP_ERROR;
    }

    // Отображение разделяемой памяти
    shared_data_t* shared = mmap(NULL, sizeof(shared_data_t),
                                 PROT_READ | PROT_WRITE, MAP_SHARED, shm_fd, 0);
    if (shared == MAP_FAILED) {
        perror("mmap");
        close(shm_fd);
        return MMAP_ERROR;
    }

    // Открытие выходного файла
    FILE* output = fopen(output_name, "w");
    if (output == NULL) {
        perror("fopen");
        munmap(shared, sizeof(shared_data_t));
        close(shm_fd);
        return FILE_OPEN_ERROR;
    }

    printf("Дочерний процесс начал работу (SHM: %s, Output: %s)\n", shm_name,
output_name);

    // Основной цикл обработки
    while (1) {
        if (shared->process_complete) {
            break;
        }
```

```
        if (shared->data_ready) {
            // Копируем данные для обработки
            char buffer[MAX_LINE_LENGTH];
            strncpy(buffer, shared->data, MAX_LINE_LENGTH - 1);
            buffer[MAX_LINE_LENGTH - 1] = '\0';

            // Обрабатываем строку
            remove_vowels(buffer);

            // Записываем результат в файл
            if (fprintf(output, "%s\n", buffer) < 0) {
                perror("fprintf");
                fclose(output);
                munmap(shared, sizeof(shared_data_t));
                close(shm_fd);
                return IO_ERROR;
            }
            fflush(output);

            // Сбрасываем флаг готовности данных
            shared->data_ready = 0;
        }

        usleep(1000); // Небольшая пауза для уменьшения нагрузки на CPU
    }

    printf("Дочерний процесс завершил работу\n");

    // Освобождение ресурсов
    fclose(output);
    munmap(shared, sizeof(shared_data_t)); // удаление отображения
    close(shm_fd);

    return STATUS_OK;
}
```

os_utils.h

```
#ifndef OS_UTILS_H
#define OS_UTILS_H

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/wait.h>
#include <sys/mman.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <errno.h>
#include <time.h>
#include <ctype.h>
```

```c
#define MAX_LINE_LENGTH 1024
#define SHARED_MEM_SIZE 4096
#define MAX_FILENAME_LENGTH 256

typedef enum {
    STATUS_OK = 0,
    MMAP_ERROR = 1,
    FORK_ERROR = 2,
    EXEC_ERROR = 3,        // Добавлена эта строка
    INVALID_INPUT = 4,
    FILE_OPEN_ERROR = 5,
    IO_ERROR = 6
} StatusCode;

// Структура для разделяемой памяти
typedef struct {
    char data[MAX_LINE_LENGTH];
    int data_ready;        // Флаг наличия данных
    int process_complete;  // Флаг завершения процесса
} shared_data_t;

void remove_vowels(char* str);

#endif
```

Протокол работы программы

Тестирование:

```
maxva@LAPTOP-RQH0OLUE:/mnt/d/Programming/OS/lab3/src$ ./parent
Введите имя shared memory для child1: /shm_child1
Введите имя файла вывода для child1: output1.txt
Введите имя shared memory для child2: /shm_child2
Введите имя файла вывода для child2: output2.txt
Родительский процесс начался
Введите строки (для выхода введите QUIT):
Дочерний процесс начал работу (SHM: /shm_child2, Output: output2.txt)
Дочерний процесс начал работу (SHM: /shm_child1, Output: output1.txt)
Hello world
Отправлено в Child2 (вероятность: 81%)
Programming is fun
Отправлено в Child1 (вероятность: 29%)
Operating systems
Отправлено в Child1 (вероятность: 73%)
QUIT
Дочерний процесс завершил работу
Дочерний процесс завершил работу
Родительский и дочерние процессы успешно завершены
```

Strace:

$ strace -f ./parent

execve("./parent", ["./parent"], 0x7ffd39e00de8 /* 38 vars */) = 0

brk(NULL)   = 0x56178f248000

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7efe2e7f8000

access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)

openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3

fstat(3, {st_mode=S_IFREG|0644, st_size=37779, ...}) = 0

mmap(NULL, 37779, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7efe2e7ee000

close(3) = 0

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\243\2\0\0\0\0"..., 832) = 832

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0"..., 784, 64) = 784

fstat(3, {st_mode=S_IFREG|0755, st_size=2125328, ...}) = 0

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0"..., 784, 64) = 784

mmap(NULL, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7efe2e5dc000

mmap(0x7efe2e604000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7efe2e604000

mmap(0x7efe2e78c000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b0000) = 0x7efe2e78c000

mmap(0x7efe2e7db000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000) = 0x7efe2e7db000

mmap(0x7efe2e7e1000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7efe2e7e1000

close(3) = 0

mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7efe2e5d9000

arch_prctl(ARCH_SET_FS, 0x7efe2e5d9740) = 0

set_tid_address(0x7efe2e5d9a10) = 23939

set_robust_list(0x7efe2e5d9a20, 24) = 0

rseq(0x7efe2e5da060, 0x20, 0, 0x53053053) = 0

mprotect(0x7efe2e7db000, 16384, PROT_READ) = 0

mprotect(0x5617626d9000, 4096, PROT_READ) = 0

mprotect(0x7efe2e830000, 8192, PROT_READ) = 0

prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

munmap(0x7efe2e7ee000, 37779) = 0

fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x5), ...}) = 0

getrandom("\x1e\x5c\x17\xd4\x98\x68\xdc\x5d", 8, GRND_NONBLOCK) = 8

brk(NULL) = 0x56178f248000

brk(0x56178f269000) = 0x56178f269000

fstat(0, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x5), ...}) = 0

write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \320\270\320\274\321\217 shared mem"..., 51Введите имя shared memory для child1:) = 51

read(0, /shmm1

"/shmm1\n", 1024) = 7

write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \320\270\320\274\321\217 \321\204\320\260\320\271\320\273\320\260"..., 61Введите имя файла вывода для child1:) = 61

read(0, ff1.txt

"ff1.txt\n", 1024) = 8

write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \320\270\320\274\321\217 shared mem"..., 51Введите имя shared memory для child2:) = 51

read(0, /shmm2

"/shmm2\n", 1024) = 7

write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265 \320\270\320\274\321\217 \321\204\320\260\320\271\320\273\320\260"..., 61Введите имя файла вывода для child2:) = 61

read(0, ff2.txt

"ff2.txt\n", 1024) = 8

openat(AT_FDCWD, "/dev/shm/shmm1", O_RDWR|O_CREAT|O_NOFOLLOW|O_CLOEXEC, 0666) = 3

ftruncate(3, 1032) = 0

openat(AT_FDCWD, "/dev/shm/shmm2", O_RDWR|O_CREAT|O_NOFOLLOW|O_CLOEXEC, 0666) = 4

ftruncate(4, 1032) = 0

mmap(NULL, 1032, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0) = 0x7efe2e7f7000

mmap(NULL, 1032, PROT_READ|PROT_WRITE, MAP_SHARED, 4, 0) = 0x7efe2e7f6000

clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLDstrace: Process 24163 attached

, child_tidptr=0x7efe2e5d9a10) = 24163

[pid 24163] set_robust_list(0x7efe2e5d9a20, 24 <unfinished ...>

[pid 23939] clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD <unfinished ...>

[pid 24163] <... set_robust_list resumed>) = 0

strace: Process 24164 attached

[pid 24163] execve("./child", ["child", "/shmm1", "ff1.txt"], 0x7ffd2f880438 /* 38 vars */ <unfinished ...>

[pid 23939] <... clone resumed>, child_tidptr=0x7efe2e5d9a10) = 24164

[pid 24164] set_robust_list(0x7efe2e5d9a20, 24 <unfinished ...>

[pid 23939] write(1, "\320\240\320\276\320\264\320\270\321\202\320\265\320\273\321\214\321\201\320\272\320\270\320\271 \320\277\321\200\320\276\321"..., 55 <unfinished ...>

Родительский процесс начался

[pid 24164] <... set_robust_list resumed>) = 0

[pid 23939] <... write resumed>) = 55

[pid 24164] execve("./child", ["child", "/shmm2", "ff2.txt"], 0x7ffd2f880438 /* 38 vars */ <unfinished ...>

[pid 23939] write(1, "\320\222\320\262\320\265\320\264\320\270\321\202\320\265
\321\201\321\202\321\200\320\276\320\272\320\270 (\320\264\320"..., 71Введите строки (для выхода введите
QUIT):

) = 71

[pid 23939] read(0, <unfinished ...>

[pid 24163] <... execve resumed>) = 0

[pid 24163] brk(NULL) = 0x5601e8af3000

[pid 24164] <... execve resumed>) = 0

[pid 24164] brk(NULL <unfinished ...>

[pid 24163] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 24164] <... brk resumed>) = 0x55cddcb2b000

[pid 24163] <... mmap resumed>) = 0x7f6ae0457000

[pid 24163] access("/etc/ld.so.preload", R_OK <unfinished ...>

[pid 24164] mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 24163] <... access resumed>) = -1 ENOENT (No such file or directory)

[pid 24164] <... mmap resumed>) = 0x7f31ccfa7000

[pid 24163] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 24164] access("/etc/ld.so.preload", R_OK <unfinished ...>

[pid 24163] <... openat resumed>) = 3

[pid 24164] <... access resumed>) = -1 ENOENT (No such file or directory)

[pid 24163] fstat(3, <unfinished ...>

[pid 24164] openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 24163] <... fstat resumed>{st_mode=S_IFREG|0644, st_size=37779, ...}) = 0

[pid 24164] <... openat resumed>) = 3

[pid 24163] mmap(NULL, 37779, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>

[pid 24164] fstat(3, <unfinished ...>

[pid 24163] <... mmap resumed>) = 0x7f6ae044d000

[pid 24164] <... fstat resumed>{st_mode=S_IFREG|0644, st_size=37779, ...}) = 0

[pid 24163] close(3 <unfinished ...>

[pid 24164] mmap(NULL, 37779, PROT_READ, MAP_PRIVATE, 3, 0 <unfinished ...>

[pid 24163] <... close resumed>) = 0

[pid 24164] <... mmap resumed>) = 0x7f31ccf9d000

[pid 24163] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC <unfinished
...>

[pid 24164] close(3 <unfinished ...>

[pid 24163] <... openat resumed>) = 3

[pid 24164] <... close resumed>) = 0

[pid 24163] read(3, <unfinished ...>

[pid 24164] openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC <unfinished ...>

[pid 24163] <... read resumed>"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\243\2\0\0\0\0"..., 832) = 832

[pid 24164] <... openat resumed>) = 3

[pid 24163] pread64(3, <unfinished ...>

[pid 24164] read(3, <unfinished ...>

[pid 24163] <... pread64 resumed>"\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

[pid 24164] <... read resumed>"\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\243\2\0\0\0\0"..., 832) = 832

[pid 24163] fstat(3, <unfinished ...>

[pid 24164] pread64(3, <unfinished ...>

[pid 24163] <... fstat resumed>{st_mode=S_IFREG|0755, st_size=2125328, ...}) = 0

[pid 24164] <... pread64 resumed>"\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

[pid 24163] pread64(3, <unfinished ...>

[pid 24164] fstat(3, <unfinished ...>

[pid 24163] <... pread64 resumed>"\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

[pid 24164] <... fstat resumed>{st_mode=S_IFREG|0755, st_size=2125328, ...}) = 0

[pid 24163] mmap(NULL, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0 <unfinished ...>

[pid 24164] pread64(3, <unfinished ...>

[pid 24163] <... mmap resumed>) = 0x7f6ae023b000

[pid 24164] <... pread64 resumed>"\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

[pid 24163] mmap(0x7f6ae0263000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000 <unfinished ...>

[pid 24164] mmap(NULL, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0 <unfinished ...>

[pid 24163] <... mmap resumed>) = 0x7f6ae0263000

[pid 24164] <... mmap resumed>) = 0x7f31ccd8b000

[pid 24163] mmap(0x7f6ae03eb000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b0000 <unfinished ...>

[pid 24164] mmap(0x7f31ccdb3000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000 <unfinished ...>

[pid 24163] <... mmap resumed>) = 0x7f6ae03eb000

[pid 24164] <... mmap resumed>) = 0x7f31ccdb3000

[pid 24163] mmap(0x7f6ae043a000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000 <unfinished ...>

[pid 24164] mmap(0x7f31ccf3b000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b0000) = 0x7f31ccf3b000

[pid 24163] <... mmap resumed>) = 0x7f6ae043a000

[pid 24164] mmap(0x7f31ccf8a000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000 <unfinished ...>

[pid 24163] mmap(0x7f6ae0440000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 24164] <... mmap resumed>) = 0x7f31ccf8a000

[pid 24163] <... mmap resumed>) = 0x7f6ae0440000

[pid 24164] mmap(0x7f31ccf90000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 24163] close(3) = 0

[pid 24164] <... mmap resumed>) = 0x7f31ccf90000

[pid 24163] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 24164] close(3 <unfinished ...>

[pid 24163] <... mmap resumed>) = 0x7f6ae0238000

[pid 24164] <... close resumed>) = 0

[pid 24163] arch_prctl(ARCH_SET_FS, 0x7f6ae0238740 <unfinished ...>

[pid 24164] mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0 <unfinished ...>

[pid 24163] <... arch_prctl resumed>) = 0

[pid 24164] <... mmap resumed>) = 0x7f31ccd88000

[pid 24163] set_tid_address(0x7f6ae0238a10 <unfinished ...>

[pid 24164] arch_prctl(ARCH_SET_FS, 0x7f31ccd88740 <unfinished ...>

[pid 24163] <... set_tid_address resumed>) = 24163

[pid 24164] <... arch_prctl resumed>) = 0

[pid 24163] set_robust_list(0x7f6ae0238a20, 24 <unfinished ...>

[pid 24164] set_tid_address(0x7f31ccd88a10 <unfinished ...>

[pid 24163] <... set_robust_list resumed>) = 0

[pid 24164] <... set_tid_address resumed>) = 24164

[pid 24163] rseq(0x7f6ae0239060, 0x20, 0, 0x53053053 <unfinished ...>

[pid 24164] set_robust_list(0x7f31ccd88a20, 24 <unfinished ...>

[pid 24163] <... rseq resumed>) = 0

[pid 24164] <... set_robust_list resumed>) = 0

[pid 24163] mprotect(0x7f6ae043a000, 16384, PROT_READ <unfinished ...>

[pid 24164] rseq(0x7f31ccd89060, 0x20, 0, 0x53053053 <unfinished ...>

[pid 24163] <... mprotect resumed>)   = 0

[pid 24164] <... rseq resumed>)   = 0

[pid 24163] mprotect(0x5601be10c000, 4096, PROT_READ) = 0

[pid 24164] mprotect(0x7f31ccf8a000, 16384, PROT_READ <unfinished ...>

[pid 24163] mprotect(0x7f6ae048f000, 8192, PROT_READ) = 0

[pid 24164] <... mprotect resumed>)   = 0

[pid 24163] prlimit64(0, RLIMIT_STACK, NULL, <unfinished ...>

[pid 24164] mprotect(0x55cdd3e53000, 4096, PROT_READ <unfinished ...>

[pid 24163] <... prlimit64 resumed>{rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

[pid 24164] <... mprotect resumed>)   = 0

[pid 24163] munmap(0x7f6ae044d000, 37779 <unfinished ...>

[pid 24164] mprotect(0x7f31ccfdf000, 8192, PROT_READ <unfinished ...>

[pid 24163] <... munmap resumed>)   = 0

[pid 24164] <... mprotect resumed>)   = 0

[pid 24163] openat(AT_FDCWD, "/dev/shm/shmm1", O_RDWR|O_NOFOLLOW|O_CLOEXEC <unfinished
...>

[pid 24164] prlimit64(0, RLIMIT_STACK, NULL, <unfinished ...>

[pid 24163] <... openat resumed>)   = 3

[pid 24164] <... prlimit64 resumed>{rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

[pid 24163] mmap(NULL, 1032, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0 <unfinished ...>

[pid 24164] munmap(0x7f31ccf9d000, 37779 <unfinished ...>

[pid 24163] <... mmap resumed>)   = 0x7f6ae0456000

[pid 24164] <... munmap resumed>)   = 0

[pid 24163] getrandom(<unfinished ...>

[pid 24164] openat(AT_FDCWD, "/dev/shm/shmm2", O_RDWR|O_NOFOLLOW|O_CLOEXEC <unfinished
...>

[pid 24163] <... getrandom resumed>"\xc2\xda\x1f\x4f\x92\xb1\x0b\x66", 8, GRND_NONBLOCK) = 8

[pid 24164] <... openat resumed>)   = 3

[pid 24163] brk(NULL <unfinished ...>

[pid 24164] mmap(NULL, 1032, PROT_READ|PROT_WRITE, MAP_SHARED, 3, 0 <unfinished ...>

[pid 24163] <... brk resumed>)   = 0x5601e8af3000

[pid 24164] <... mmap resumed>)   = 0x7f31ccfa6000

[pid 24163] brk(0x5601e8b14000 <unfinished ...>

[pid 24164] getrandom(<unfinished ...>

[pid 24163] <... brk resumed>)   = 0x5601e8b14000

[pid 24164] <... getrandom resumed>"\x48\xc6\xc9\xb1\x56\x88\xf3\x81", 8, GRND_NONBLOCK) = 8

[pid 24163] openat(AT_FDCWD, "ff1.txt", O_WRONLY|O_CREAT|O_TRUNC, 0666 <unfinished ...>

[pid 24164] brk(NULL)   = 0x55cddcb2b000

[pid 24164] brk(0x55cddcb4c000)   = 0x55cddcb4c000

[pid 24164] openat(AT_FDCWD, "ff2.txt", O_WRONLY|O_CREAT|O_TRUNC, 0666 <unfinished ...>

[pid 24163] <... openat resumed>)   = 4

[pid 24163] fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x5), ...}) = 0

[pid 24163] write(1, "\320\224\320\276\321\207\320\265\321\200\320\275\320\270\320\271
\320\277\321\200\320\276\321\206\320\265\321\201\321\201 "..., 87Дочерний процесс начал работу (SHM:
/shmm1, Output: ff1.txt)

) = 87

[pid 24164] <... openat resumed>)   = 4

[pid 24164] fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0x5), ...}) = 0

[pid 24164] write(1, "\320\224\320\276\321\207\320\265\321\200\320\275\320\270\320\271
\320\277\321\200\320\276\321\206\320\265\321\201\321\201 "..., 87Дочерний процесс начал работу (SHM:
/shmm2, Output: ff2.txt)

) = 87

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if
SA_RESTART is set)

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if
SA_RESTART is set)

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if
SA_RESTART is set)

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, 0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, 0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, 0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, <unfinished ...>

```
[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if
SA_RESTART is set)

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0,  <unfinished ...>

[pid 24164] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] <... read resumed>0x56178f2486b0, 1024) = ? ERESTARTSYS (To be restarted if
SA_RESTART is set)

[pid 24163] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] --- SIGWINCH {si_signo=SIGWINCH, si_code=SI_KERNEL} ---

[pid 23939] read(0, Hello world

"Hello world\n", 1024) = 12

[pid 23939] write(1,
"\320\236\321\202\320\277\321\200\320\260\320\262\320\273\320\265\320\275\320\276 \320\262 Child1 ("...,
61Отправлено в Child1 (вероятность: 30%)

) = 61

[pid 24163] fstat(4, {st_mode=S_IFREG|0777, st_size=0, ...}) = 0

[pid 24163] write(4, "Hll wrld\n", 9)   = 9

[pid 23939] read(0, Programming

"Programming\n", 1024) = 12

[pid 23939] write(1,
"\320\236\321\202\320\277\321\200\320\260\320\262\320\273\320\265\320\275\320\276 \320\262 Child1 ("...,
61Отправлено в Child1 (вероятность: 49%)

) = 61

[pid 24163] write(4, "Prgrmmng\n", 9)   = 9

[pid 23939] read(0, Yes

"Yes\n", 1024)      = 4

[pid 23939] write(1,
"\320\236\321\202\320\277\321\200\320\260\320\262\320\273\320\265\320\275\320\276 \320\262 Child1 ("...,
61Отправлено в Child1 (вероятность: 74%)

) = 61

[pid 24163] write(4, "s\n", 2)          = 2

[pid 23939] read(0, Nooooope

"Nooooope\n", 1024) = 9

[pid 23939] write(1,
"\320\236\321\202\320\277\321\200\320\260\320\262\320\273\320\265\320\275\320\276 \320\262 Child1 ("...,
61Отправлено в Child1 (вероятность: 21%)

) = 61

[pid 24163] write(4, "Np\n", 3)         = 3
```

[pid 23939] read(0, Asssss

"Asssss\n", 1024) = 7

[pid 23939] write(1, "\320\236\321\202\320\277\321\200\320\260\320\262\320\273\320\265\320\275\320\276 \320\262 Child1 ("..., 60Отправлено в Child1 (вероятность: 9%)

) = 60

[pid 24163] write(4, "sssss\n", 6) = 6

[pid 23939] read(0, QUIT

"QUIT\n", 1024) = 5

[pid 23939] wait4(24163, <unfinished ...>

[pid 24164] write(1, "\320\224\320\276\321\207\320\265\321\200\320\275\320\270\320\271 \320\277\321\200\320\276\321\206\320\265\321\201\321\201 "..., 62 <unfinished ...>

[pid 24163] write(1, "\320\224\320\276\321\207\320\265\321\200\320\275\320\270\320\271 \320\277\321\200\320\276\321\206\320\265\321\201\321\201 "..., 62Дочерний процесс завершил работу

Дочерний процесс завершил работу

 <unfinished ...>

[pid 24164] <... write resumed>) = 62

[pid 24163] <... write resumed>) = 62

[pid 24164] close(4 <unfinished ...>

[pid 24163] close(4 <unfinished ...>

[pid 24164] <... close resumed>) = 0

[pid 24164] munmap(0x7f31ccfa6000, 1032) = 0

[pid 24164] close(3 <unfinished ...>

[pid 24163] <... close resumed>) = 0

[pid 24164] <... close resumed>) = 0

[pid 24163] munmap(0x7f6ae0456000, 1032 <unfinished ...>

[pid 24164] exit_group(0 <unfinished ...>

[pid 24163] <... munmap resumed>) = 0

[pid 24163] close(3 <unfinished ...>

[pid 24164] <... exit_group resumed>) = ?

[pid 24163] <... close resumed>) = 0

[pid 24163] exit_group(0) = ?

[pid 24164] +++ exited with 0 +++

[pid 23939] <... wait4 resumed>NULL, 0, NULL) = ? ERESTARTSYS (To be restarted if SA_RESTART is set)

[pid 23939] --- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=24164, si_uid=1000, si_status=0, si_utime=5967 /* 59.67 s */, si_stime=1 /* 0.01 s */} ---

[pid 23939] wait4(24163, <unfinished ...>

[pid 24163] +++ exited with 0 +++

<... wait4 resumed>NULL, 0, NULL)   = 24163

--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=24163, si_uid=1000, si_status=0, si_utime=5967 /* 59.67 s */, si_stime=1 /* 0.01 s */} ---

wait4(24164, NULL, 0, NULL)   = 24164

write(1, "\320\240\320\276\320\264\320\270\321\202\320\265\320\273\321\214\321\201\320\272\320\270\320\271\320\270 \320\264\320\276"..., 96Родительский и дочерние процессы успешно завершены

) = 96

munmap(0x7efe2e7f7000, 1032)   = 0

munmap(0x7efe2e7f6000, 1032)   = 0

close(3)   = 0

unlink("/dev/shm/shmm1")   = 0

close(4)   = 0

unlink("/dev/shm/shmm2")   = 0

exit_group(0)   = ?

+++ exited with 0 +++

Вывод

В ходе выполнения лабораторной работы были освоены принципы межпроцессного взаимодействия с использованием технологии File Mapping (Memory-Mapped Files). На практике реализован механизм разделяемой памяти POSIX, позволяющий нескольким процессам обмениваться данными через общую область памяти без необходимости копирования.